REMARKS

I. INTRODUCTION

In response to the Office Action dated October 19, 2005, claims 1, 12, 25, 38, 51, 64, and 77 have been amended. Claims 1-11, 25-37, 51-63, 77-89 are withdrawn from consideration, and claims 12-24, 38-50, 64-76 remain pending under consideration. Entry of these amendments, and re-consideration of the application, as amended, is requested.

II. INTERVIEW SUMMARY

Applicants appreciate the time and consideration taken by the Examiner and his supervisor to review and discuss the case.

On December 16, 2006, an interview was conducted between Jason S. Feldmar, Steven Guttman, Examiner Betit, and Examiner Betit's supervisor. Discussion of the current claims in view of the cited prior art was discussed. Agreement was not reached with respect to allowable subject matter.

III. RESTRICTION REQUIREMENT

The Restriction Requirement asserts that the three claim groups are distinct because they are directed to a self-expanding data package, generating data in a self-expanding data package, and utilizing data in a self-expanding data package.

Applicants assert that restriction is improper and therefore traverse the requirement. Applicants note that claim 1 provides for a self-expanding data package containing a set of constant lists, row validation calculations, and expanding the package using such constant lists and calculations. Similarly, claim 12 provides for generating the self-expanding data package by stating what is in the data package - namely, the set of constant lists, the calculations and the ability to expand the data package. In this regard, claim 12 provides for generating the various values in the set of constant lists and generating the calculations. Further, claim 12 provides for expanding the package using the constant lists and calculations. Similar to both claims 1 and 12, claim 25 provides for utilizing data in the self-expanding data package by receiving the package containing values in a set of constant lists and calculations. Further, claim 25 provides for expanding the package using the constant list and calculations.

Thus, all of the independent claims relate to and provide very similar limitations. Even more specifically, Applicants submit that claim 12 is the process/method for making the product/table of claim 1. Accordingly, under MPEP 821.04, Applicants are entitled to rejoinder. Further, claim 25 has nearly identical limitations and would clearly not require any further search and/or consideration. In addition, the inventions are clearly related in their operation and effect and are therefore not distinct under MPEP 802.01.

In this regard, Applicants urge the Examiner take into consideration that the subject matter of each of the claim Groups is linked by a common inventive concept. According to M.P.E.P. §821.04 and 03, there are two criteria for a proper restriction requirement. First, the two inventions must be independent and distinct. In addition, there must be a serious burden on the Examiner if restriction is not required. Even if the first criterion has been met in the present case, which it has not, the second criterion has not been met.

In view of the above, Applicants respectfully request rejoinder of the non-elected claims.

IV. PRIOR ART REJECTIONS

In paragraphs (3)-(4) of the Office Action, claims 12-16, 18-24, 38-42, 44-50, 64-68, and 70-76 were rejected under 35 U.S.C. §103(a) as being unpatentable over "Complex Queries in XML-GL" (XML) in view of Williams, U.S. Patent No. 6,591,272 (Williams). In paragraph (5) of the Office Action, claims 17, 43, and 69 were rejected under 35 U.S.C. §103(a) as being unpatentable over XML in view of Williams as applied to claims 12-16, 18-24, 38-42, 44-50, 64-68, and 70-76, and further in view of McClendon et al., U.S. Patent No. 6,625,619 (McClendon).

Applicants respectfully traverse these rejections.

The independent claims were rejected as follows:

> As to claim 1, "Complex Queries in XML-GL" teaches a method for generating data in a self-expanding data package in a computer system comprising:
>
> generating one or more values in a set of one or more constant lists and storing said one or more values in the self-expanding data package (see section 2. Preliminary Overview of XML-GL);
>
> generating one or more calculations that operate on one or more values in the set of one or more constant lists (see section 3. Simple Queries and see section 4. Complex Queries);
>
> expanding the self-expanding data package into an expanded table having expanded table rows, by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values (see section 4.2 Cartesian Product).
>
> "Complex Queries in XML-GL" does not
>
> (a) teach a computer system, and

(b) storing said one more calculations in the self-expanding data package and transmitting the self-expanding data package to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows.

Williams teaches this (a), see figure 1, reference number 12, and (b), see column 4, line 60 through column 5, line 29. Therefore, it would have been obvious for a person of ordinary skill in the art at the time the invention was made to have modified "Complex Queries in XML-GL" to include the teachings of Williams because they would allow the client (second) computer to assemble final objects from pseudo-objects and metadata into the format required by the software on the client computer (see Williams, column 5, lines 25-29).

...

As to claim 38, "Complex Queries in XML-GL" teaches an apparatus for generating data in a self-expanding data package in a computer system comprising:

(b) generating a self-expanding data package and storing the self-expanding data package, wherein the self-expanding data package comprising:

(i) one or more values in a set of one or more constant lists; and

(ii) one or more calculations that operate on one or more values in the set of one or more constant lists;

expanding the self-expanding data package into an expanded table having expanded table rows, by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values. "Complex Queries in XML-GL" does not teach:

(a) a computer system having a memory and a data storage device coupled thereto; one or more computer programs, performed by the computer system, and storing the self-expanding data package in the memory; and

(b) the self-expanding data package is transmitted to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows.

Williams teaches this (a), see column 4, lines 48-59, and (b), see column 4, line 60 through column 5, line 29. Therefore, it would have been obvious for a person of ordinary skill in the art at the time the invention was made to have modified "Complex Queries in XML-GL" to include the teachings of Williams because they would allow the client (second) computer to assemble final objects from pseudo-objects and metadata into the format required by the software on the client computer (see Williams, column 5, lines 25-29).

...

As to claim 64, "Complex Queries in XML-GL" teaches an article of manufacture comprising a program storage medium readable by a computer and embodying one or more instructions executable by the computer to perform a method for generating data in a self-expanding data package in a computer system, the method comprising:

generating, in the self-expanding data package, one or more values in a set of one or more constant lists;

generating, in the self-expanding data package, one or more calculations that [ean] operate on one or more values in the set of one or more constant lists;

expanding the self-expanding data package into an expanded table having expanded table rows, by combining every value in each constant list with any combination of values from remaining parameters and performing the one or more calculations on the one or more values.

"Complex Queries in XML-GL" does not teach:

(a) an article of manufacture comprising a program storage medium readable by a computer and embodying one or more instructions executable by the computer, and

(b) wherein the self-expanding data package is transmitted to a second computer system that expands the self-expanding data package into an expanded table having expanded table rows.

Williams teaches this (a), see column 4, lines 48-50, and (b), see column 4, line 60 through column 5, line 29. Therefore, it would have been obvious for a person of ordinary skill in the art at the time the invention was made to have modified "Complex Queries in XML-

> GL" to include the teachings of Williams because they would allow the client (second) computer to assemble final objects from pseudo-objects and metadata into the format required by the software on the client computer (see Williams, column 5, lines 25-29).

Applicants traverse the above rejections for one or more of the following reasons:

(1) Neither XML, Williams, and McClendon teach, disclose or suggest a single package that contains both a set of constant lists and calculations that are performed on combinations of the constant lists;

(2) Neither XML, Williams, and McClendon teach, disclose or suggest such calculations that eliminate rows/combinations of such constant lists; and

(3) There is no motivation to combine XML with Williams.

Independent claims 12, 38, and 64 are generally directed to the generation of a self-expanding data package. Specifically, values in a set of constant lists are generated and stored in the data package. In addition, calculations (that operate on the values) are generated and stored in the data package. Once the values and calculations are stored in the data package, the data package is transmitted to a second computer system that expands the data package. The claim limitations provide that the package is expanded into a table having rows. In addition, the expansion is performed by combining each value with other parameters (i.e., in the data package) and performing the calculations (from the data package) on the values. As amended, each expanded row of the table comprises one of the combinations. Further, as amended, the calculations eliminate one or more rows from the table. Accordingly, all of the information for expanding the data package is contained within the data package itself. In other words, the values for the set of constant lists and the calculations performed on the values are both generated and then stored in the data package.

Contrary to the above-claimed functionality, the prior art fails to teach the invention in multiple respects. Firstly, it is noted that both the constant lists and the calculations are stored in the same self-expanding data package. Further, the claims specifically provide for how to expand the data in the package into a table having multiple rows. Namely, the constant lists are all combined with each other and each combination is represented in a row of the table. In addition, the calculations are configured to remove/eliminate one or more of the rows. Such a package and capabilities of the content in the package are not even remotely contemplated, implicitly or explicitly, by any of the cited references.

Again, a claimed aspect of the claims is the storage of both the constant lists and the calculations in the same data package. The Office Action recites the XML reference for the generation of the values and the generation of the calculations. However, XML-GL reference merely describes a graphical query language and the ability to construct various queries in a graphical manner. Notoriously missing from the XML reference is any description of storing both the constant lists and calculations in the same data package that is transmitted to a second computer system where it is expanded as claimed. In this regard, Applicants submit that it would be contrary to the teaching of XML-GL to store XML-GL's queries into a package - not to mention that storing such queries with the data itself in a single location/package is not even hinted at. Again, the Cartesian product section of XML merely describes a particular type of product/query. However, Applicants are not asserting that queries or Cartesian products are novel. Instead, Applicants are asserting the unique combination of elements as set forth in the claims are novel. The mere capability to create a query or Cartesian product does not remotely teach, disclose, suggest, hint, or allude to the storage of the constant lists and calculations as claimed.

In addition to the above, the Office Action relies on XML to teach the storage of the constant lists in the package. Applicants acknowledge that merely storing information in a table is within the prior art: However, such storage is not what is claimed. Instead, the constant lists and calculations are both stored in the same data package. The art completely fails to teach such a combined storage.

Applicants further note that the calculations serve to eliminate certain rows of a resulting table. Nowhere in XML is such a teaching taught or implied.

In fact, the Office Action admits that XML-GL fails to teach the storage of the calculations in the data package. To teach the storage of such calculations, the Office Action relies on Williams. Specifically, the Action relies on Williams col. 4, line 60-column 5, line 29. Applicants respectfully traverse the rejection. Col. 4, line 60-col. 5, line 29 provide:

> An inexperienced user can, if so desired, easily select a subset of all possible objects represented by the databases through use of a simple and intuitive graphical interface. Conversely, tables and interrelationships not required by the application can be easily deselected through use of the said simple and intuitive graphical interface.
>
> Source code for the classes is then generated from the standardized view when merged with the prepared template definitions. The source code is then compiled into binary executable form into the classes desired. Pseudo-objects are then produced by dynamic generation and execution of pre-optimized SQL, enveloping values that result from execution of the generated prepared SQL statements. Result sets from said associated prepared statement operations from the appropriate database tables and rows are normalized into a standard format, then combined with metadata from

> the database schemas. The pseudo-objects are then ready for transmission to the client computer or the requester of the objects desired. The present invention also relates to a method of communicating elements of a database table between a server computer and a client computer. A pseudo-object is generated by the server computer with the pseudo-object comprising rows from singular database tables, or optimized joins between multiple related database tables, that comprise the object desired. The plurality of datatypes present in the relational databases are normalized into a singular standardized form to prepare the data for transmission to the requestor of the object. Metadata of the elements where the metadata is the relationship between the data elements is also generated by the server computer. The metadata and normalized pseudo-object data are transmitted from the server computer to the client computer in a single logical transmission.. At the client computer, the elements are assembled into the final objects from the pseudo-objects and metadata received, into the format required by the software on the client computer without runtime overhead on the database or middle-tier server computers.

As can be seen, this text merely provides for the creation of pseudo objects. However, such pseudo objects consists of rows of a table and metadata that merely depicts the relationship between the data elements/rows. The rows of the table may also merely be the result of an optimized join operation between multiple related tables.

However, what is absent from the text is any description of a single pseudo object that contains both (1) constant lists that can be used to create a table having rows and (2) calculations that serve to remove rows from such a created table. Instead, Williams merely describes storing the end result of a query in the form of an object along with metadata that provides a relationship between data. While the pseudo object may consists of rows from a table (or an optimized join), the metadata is not even remotely similar to the calculations that are stored as claimed. Again the calculations not only operate on the values in the constant lists but they eliminate rows of a table that are produced from the constant lists. In this regard, as stated in the cited text, Williams describes a method of communicating elements of a database table between a server and a client (see col. 5, lines 11-13). However, such a teaching is not even remotely similar to teaching the transmission of a various values that may be expanded into a full table as claimed.

As stated above, the various claimed elements provide for a single data package that contains both constant lists that are expanded into a table by forming various combinations (with each row representing a combination), and calculations that eliminate one or more such combinations/rows. No such invention is taught, disclosed, or suggested, either implicitly or explicitly, in any of the cited references.

In addition to the above, Applicants submit that there is no motivation to combine Williams with XML. XML-GL provides the ability to graphically represent a query of an XML document (see XML Abstract). However, Williams relates to making and transmitting objects from a database

server computer to a client computer (see title). There is no hint or suggestion in either reference that alludes to the ability to use complex graphical language queries in the form of XML-GL with the transmission of object methods of Williams. The motivation provided in the Office Action was that it would allow the client computer to assemble final objects from pseudo-objects and metadata into the format required by the software on the client computer. However, the ability to assemble objects from pseudo objects into a format desired as provided in Williams does not even remotely suggest the use of a graphical language query system as described in XML-GL. The ability to retrieve and restructure XML documents in a simple and intuitive way (XML-GL Abstract) has no impact and no connection to assembling objects from pseudo objects in a desired format as set forth in Williams.

Under MPEP 2143, it is the Examiner's obligation to set forth a prima facie case of obviousness. As part of establishing the case, the Examiner must meet three criteria: he must show that some suggestion or motivation, either in the references themselves or in the knowledge generally avail-able to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

None of the three criteria have been established by the Examiner. Again, there is no motivation to combine the references. The mere reference to placing information into a format required on a client computer without runtime overhead on a database or middle-tier server does not even remotely suggest the combination with XML-GL. Further, such a disclosure in Williams actually serves to teach away from combining with XML-GL. In this regard, Williams point is to place the information in the desired format WITHOUT runtime overhead on a database. However, the XML-GL reference would actually create more overhead by forcing the user to utilize a graphical query language. Further, the graphical query including the Cartesian product operation would force multiple query operations and increase the overhead on the database. In addition, there is no reasonable expectation of success. There is no knowledge or indication that XML-GL queries can be transmitted in the manner specified in Williams. Further, the references, either alone or in

combination, fail to teach all of the claim limitations. Specifically, the limitations relating to a single data package that contains the constant lists and the calculations that eliminate rows of the table resulting from combinations of the constant lists.

In view of the above, Applicants submit that when combined, the references actually teach away from Applicants' invention in that they are inoperative and will likely fail to work.

In addition, the other cited references fail to cure the deficiencies described above.

Further, the various elements of Applicants' claimed invention together provide operational advantages over the systems disclosed in XML, Williams, and McClendon. In addition, Applicants' invention solves problems not recognized by XML, Williams, and McClendon.

Thus, Applicants submit that independent claims 12, 38, and 64 are allowable over XML, Williams, and McClendon. Further, dependent claims 13-24, 39-50, and 65-74 are submitted to be allowable over XML, Williams, and McClendon in the same manner, because they are dependent on independent claims 12, 38, and 64, respectively, and because they contain all the limitations of the independent claims. In addition, dependent claims 13-24, 39-50, and 65-74 recite additional novel elements not shown by XML, Williams, and McClendon.

V. CONCLUSION

In view of the above, it is submitted that this application is now in good order for allowance and such allowance is respectfully solicited. Should the Examiner believe minor matters still remain that can be resolved in a telephone interview, the Examiner is urged to call Applicants' undersigned attorney.

Respectfully submitted,

Craig Storms et al.

By their attorneys,

GATES & COOPER LLP

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: December 19, 2005

By: _____
Name: Jason S. Feldmar
Reg. No.: 39,187